IN THE CLAIMS:

1. (Previously Presented) A method, comprising:

receiving a software module, the software module including references to locations within the software module, at least some of the references being backward references; and

reordering components of the software module to remove at least some of the backward references.

2. (Previously Presented) The method according to claim 1, further comprising:

adjusting at least one of the references in the software module to reflect the reordering of the components of the software module, so that the at least one of the references remains a reference to the same component, but to the component's new, reordered location, the new, reordered location coming after the at least one reference in the software module.

3. (Previously Presented) The method according to claim 2,

wherein the software module includes a symbol table, the symbol table including backward references when the reordering of the components of the software module and adjusting the at least one of the references have been completed.

4. (Previously Presented) The method according to claim 2,

wherein the software module includes a symbol table, the software module including no backward references in locations before the symbol table when the reordering of the components of the software module and adjusting the at least one of the references have been completed.

5. (Previously Presented) The method according to claim 2,

wherein the software module is a relocatable object code module in ELF format when the reordering the components of the software module and adjusting the at least one of the references have been completed.

6. (Previously Presented) The method according to claim 5,

wherein, when the software module is received, the software module is a relocatable object code module in ELF format, and

wherein, when the reordering the components of the software module and adjusting the at least one of the references have been completed, the software module includes a symbol table, the symbol table including backward references, and the software module includes no backward references from locations before the symbol table.

7. (Previously Presented) The method according to claim 1,

wherein the software module comprises at least one segment, each at least one segment comprising at least one section, and

wherein sections in the same segment are contiguously located in the software module when the reordering of the components of the software module has been completed.

8. (Original) The method according to claim 1,

wherein, when the software module is received, the software module is a relocatable object code module in ELF format.

9. (Previously Presented) A system, comprising:

a reorder module configured to receive a software module including references to locations within the software module, at least some of the references being backward references, the reorder module configured to reorder components of the software module and remove at least some of the backward references.

10. (Previously Presented) The system according to claim 9,

wherein the reorder module is configured to adjust a reference in the software module to reflect the reordering of the components of the module.

11. (Original) The system according to claim 9,

wherein the software module includes a symbol table, and

wherein the reorder module is configured not to remove backward references from the symbol table.

12. (Original) The system according to claim 9,

wherein the software module includes a symbol table, and

wherein the reorder module is configured to remove all backward references from locations before the symbol table in the reordered software module.

13. (Original) The system according to claim 9,

wherein the software module includes at least one segment, each of the at least one segments including at least one section, and the reorder module is configured to locate sections in the same segment contiguously in the reordered software module.

14. (Original) The system according to claim 9,

wherein the software module is a relocatable object code module in ELF format, and the reordered software module is a relocatable object code module in ELF format.

15. (Previously Presented) The system according to claim 14,

wherein the software module includes a symbol table,

wherein the reorder module is configured to adjust a reference in the software module to reflect the reordering of the components of the module,

wherein the reorder module is configured to remove all backward references from locations before the symbol table, and

wherein the reorder module is configured not to remove backward references from the symbol table.

16. (Previously Presented) A method, comprising

receiving a software module sequentially, the software module having at least one symbol reference;

linking the software module onto a target memory space; and

resolving the at least one symbol reference without storing the entire software module in local memory while the symbol reference is resolved.

17. (Original) The method according to claim 16, further comprising:

storing section identification information in local memory while the at least one symbol reference is resolved,

wherein the software module includes at least one section and the section identification information uniquely identifies said at least one section.

18. (Original) The method according to claim 16, further comprising:

storing symbol information in local memory, wherein said symbol information is contained in the software module.

19. (Original) The method according to claim 16,

wherein the software module includes a data section, and the data section is not stored in local memory while the at least one symbol reference is resolved.

20. (Original) The method according to claim 16,

wherein the software module includes a text section, and the text section is not stored in local memory while the at least one symbol reference is resolved.

21. (Original) The method according to claim 16,

wherein the software module is a relocatable object code module in ELF format.

22. (Original) The method according to claim 21, further comprising:

storing section identification information in local memory while the at least one symbol reference is resolved, wherein the software module includes at least one section and the section identification information uniquely identifies said at least one section; and

storing symbol information in local memory while the at least one symbol reference is resolved, wherein the symbol information is contained in the software module,

wherein the software module includes a data section, and the data section is not stored in local memory while the at least one symbol reference is resolved.

23. (Original) A system, comprising:

a linker configured to sequentially receive a software module having at least one symbol reference, the linker configured to resolve the symbol reference, the linker configured to store less than the entire software module in local memory during the resolution of the at least one symbol reference.

24. (Original) The system according to claim 23,

wherein the linker is configured to store section information in local memory while the linker resolves at least one symbol reference, and

wherein the software module sequentially received by the linker includes at least one section, and

wherein the section identification information stored by the linker uniquely identifies said at least one section.

25. (Original) The system according to claim 23,

wherein the linker is configured to store symbol information in local memory while the linker resolves the at least one symbol reference, and

wherein the symbol information is contained in the software module received by the linker.

26. (Original) The system according to claim 23,

wherein the software module received by the linker includes a data section, and wherein the linker is configured not to store the data section in local memory while the linker resolves the at least one symbol reference.

27. (Original) The system according to claim 23,

wherein the software module received by the linker includes a text section, and

wherein the linker is configured not to store the text section in local memory while the linker resolves the at least one symbol reference.

28. (Original) The system according to claim 23, further comprising:

a system symbol table.

29. (Original) The system according to claim 28, wherein

the system symbol table includes a system symbol table entry for the at least one symbol reference, the system symbol table entry including a field indicative of a defining software module which defines the at least one symbol reference.

30. (Original) The system according to claim 23, further comprising:

a software module list.

31. (Original) The system according to claim 30, wherein the software module list includes

a software module list entry for the software module.

32. (Original) The system according to claim 23, further comprising:

a link status information data structure.

33. (Original) The system according to claim 32, wherein the link status information data structure includes a link status information data structure entry for the software module.

34. (Original) The system according to claim 33, further comprising:

a software module list, the software module list including a software module list entry for the software module; and

a system symbol table, the system symbol table including a system symbol table entry for the at least one symbol reference, the system symbol table entry including a field indicative of a defining software module which define the at least one symbol reference.

35. (Original) The system according to claim 23,

wherein the software module received by the linker is a relocatable object code module in ELF format.

36. (Currently Amended) A ~~tangible~~ computer readable storage medium~~, having stored thereon machine readable instructions representing a software module, said software module comprising~~ including a set of instructions representing a software module that is executable by a processor, the set of instructions operable to:

~~a symbol table, the symbol table including at least one backward reference;~~

~~at least one component of the software module located before the symbol table in the software module;~~

~~wherein none of the at least one components of the software module located before the symbol table include backward internal references~~

receiving a software module sequentially, the software module having at least one symbol reference;

linking the software module onto a target memory space; and

resolving the at least one symbol reference without storing the entire software module in local memory while the symbol reference is resolved.

37. (Currently Amended) The ~~tangible medium~~ set of instructions according to claim 36, wherein the software module is in ELF format.

38. (Previously Presented) An article of manufacture comprising a computer-readable medium having stored thereon instructions adapted to be executed by a processor, the instructions which, when executed, define a series of steps to be used to reorder a software module, said steps comprising:

receiving a software module, the software module including references to locations within the software module, at least some of the references being backward references; and

reordering the components of the software module to remove at least some of the backward references.

39. (Previously Presented) An article of manufacture comprising a computer-readable medium having stored thereon instructions adapted to be executed by a processor, the instructions which, when executed, define a series of steps to be used to control the linking of a software module, said steps comprising:

receiving a software module sequentially, the software module having at least one symbol reference;

linking the software module onto a target memory space; and

resolving the at least one symbol reference without storing the entire software module in local memory at one time.

40. (Previously Presented) The method of claim 1, wherein

the reordering of the components of the software module is completed prior to linking the software module.

41. (Previously Presented) The method of claim 40, further comprising:

linking the reordered software module.

42. (Previously Presented) The method of claim 1, further comprising:

transferring the reordered software module to a different computer system; and linking the reordered software module on the different computer system.

43. (Previously Presented) The method of claim 1, wherein the reordered components include an ELF data section.

44. (Previously Presented) The method of claim 1, wherein the reordered components include an ELF code section.

45. (Previously Presented) The method of claim 1, wherein the reordered components include an

ELF header table.

46. (Previously Presented) The method of claim 1, wherein the reordered components include an ELF entry point table.

47. (Previously Presented) The method of claim 1, wherein the reference points to a section located prior to the reference in the received software module.

48. (Previously Presented) The method of claim 47, wherein, after the software module has been reordered, the reference is changed to point at the same section, the section having been relocated to appear after the reference in the reordered software module.

49. (Previously Presented) The method of claim 1, wherein the reference points to a table located prior to the reference in the received software module.

50. (Previously Presented) The method of claim 49, wherein, after the software module has been reordered, the reference is changed to point at the same table, the table having been relocated to appear after the reference in the reordered software module.

51. (Previously Presented) The method of claim 1, wherein the reference points into a section located prior to the reference in the received software module.

52. (Previously Presented) The method of claim 51, wherein, after the software module has been reordered, the reference points into the same section, the section having been relocated to appear

after the reference in the reordered software module.

53. (Previously Presented) The method of claim 1, wherein the reference points into a table located prior to the reference in the received software module.

54. (Previously Presented) The method of claim 53, wherein, after the software module has been reordered, the reference is changed to point into the same table, the table having been relocated to appear after the reference in the reordered software module.

55. (Previously Presented) A method, comprising:

receiving a software module, the software module including components arranged in a first order, a first one of the components including a reference to a location in a second one of the components, the second one of the components preceding the first one of the components in the first order; and

arranging the components into a second order so that the second one of the components is subsequent to the first one of the components in the second order.

56. (Previously Presented) The method of claim 55, wherein the arranging occurs prior to linking the software module.

57. (Previously Presented) The method of claim 56, further comprising:

linking the software module without storing the entire software module in local memory.

58. (Previously Presented) The method of claim 57, wherein the components include an ELF table

and an ELF section.

59. (Previously Presented) The method of claim 58, wherein the order of segments within the ELF section is preserved when the section is moved to a different position in the reordered software module.

60. (Previously Presented) The method of claim 59, wherein the only backward references between different ELF components in the reordered software module are references located in the ELF symbol table.